An Envelope-Following Approach to Switching Power Converter Simulation

Jacob White, *Member, IEEE*, and Steven B. Leeb, *Student Member, IEEE*

Abstract—**Simulating the transient behavior of switching power converter circuits is computationally expensive because these circuits are clocked at a frequency whose period is orders of magnitude smaller than the time interval of interest to the designer. It is possible to reduce the simulation time without compromising much accuracy by exploiting the property that the behavior of switching converters in a given high-frequency clock cycle is similar, but not identical, to the behavior in the preceding and following cycles. In particular, the envelope of the high-frequency clock can be followed by accurately computing the circuit behavior over occasional cycles. In this paper the implementation of an envelope-following method that is particularly efficient for open-loop switching power converters with fixed clock frequencies is described, and results demonstrating the method's effectiveness are presented.**

I. Introduction

In general, switching power converter designers rely heavily on circuit simulation programs like SPICE [8] to verify the correctness and to determine the performance of their designs. These programs simulate a circuit by first constructing a system of differential equations that describes the circuit, and then solving that system numerically with a time discretization method such as backward-Euler. When applied to circuits like switching power converters, such classical circuit simulation algorithms become extraordinarily computationally expensive. This is because switching power converters use high-frequency clocks whose periods are typically orders of magnitude smaller than the time intervals of interest to a designer. The nature of the calculations used in a circuit simulator implies that to construct the solution over the time interval of interest, an accurate solution must be computed for every cycle of the high-frequency clock in the interval, and this can involve hundreds of cycles.

The infeasibility of simulating such circuits with classical techniques has led designers to explore a variety of simulation alternatives, including specialized analog computers [4]. More popular are fast approximate simulation techniques, based on treating the switching converter's switches as ideal, and the remaining circuitry as linear [2]. In addition, it is sometimes possible to further simplify the converter circuit by eliminating certain state variables that do not contribute significantly to the output of interest [7]. Approximate techniques such as these can reduce the cost of computing the behavior of a switching converter circuit over one high-frequency clock cycle to the point where it becomes computationally feasible to simulate the circuit for the hundreds of cycles needed to construct a complete transient.

Programs based on the above techniques are reasonably efficient, but they are based on idealizations of the circuits involved which may eliminate behavior that is important to a designer. In this paper we present an approach for the detailed transient simulation of switching power circuits that does not require simplifying the circuit and is much more efficient than classical SPICE-like methods when the clock period is small compared to the simulation interval. This method, referred to as envelope-following [9], exploits the property of such circuits that the node voltage waveforms over a given high-frequency clock cycle are similar to, but not exact duplicates of, the node voltage waveforms in preceding or following cycles. This suggests that it is possible to construct a solution accurate over many clock cycles by calculating the solution accurately for a few selected cycles.

In the next section, envelope-following is introduced, and in Section III we derive a simple method for computing envelopes which involves solving a sequence of two-point boundary value problems. The two-point boundary value problems are solved with a shooting or Newton method, as described in Section IV. The computations involved are explained in Section V, and their implementation in the program Nitswit along with results from using Nitswit to simulate several switching power circuits is described in Section VI. Finally, in Section VII, conclusions and acknowledgments are given.

II. Envelope-Following

Consider the simplified buck dc-dc converter circuit in Fig. 1, [2]. This circuit's behavior in steady-state is roughly that of a modulator followed by a low-pass filter. The modulator converts the input dc source into a periodic pulse waveform and the low-pass filter time-averages the pulse waveform to produce a dc voltage at the output. In the circuit in Fig. 1, the N-channel MOS transistor combined with a diode act as the modulator, and are controlled by the input clock connected to the MOS transistor's gate. The dc output voltage of the converter is given approximately by DV_{in}, where D is the duty-cycle of the input clock waveform.

The voltage waveforms in steady state for the switch and output nodes of the buck-converter of Fig. 1 were computed numerically using a standard backward-Euler integration scheme; the computed timepoints are plotted in Figs. 2 and 3 (for the simulation, the dc input was 10 V, the clock was a 100-kHz square wave and $R = 140\ \Omega$, $L = 420\ \mu$H and $C = 38\ \mu$F). As should be expected in a realistic simulation, the plots clearly indicate the effect of the finite conductivity of the MOS transistor and the nonideal behavior of the diode. More relevant to

Manuscript received July 27, 1990; revised November 5, 1990. This work was supported by the Defense Advanced Research Projects Agency contract N00014-87-K-825, the National Science Foundation contract MIP-8858764, and a grant from Digital Equipment Corporation.

The authors are with the Research Laboratory of Electronics and the Laboratory for Electromagnetic and Electronic Systems, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139.

IEEE Log Number 9042332.

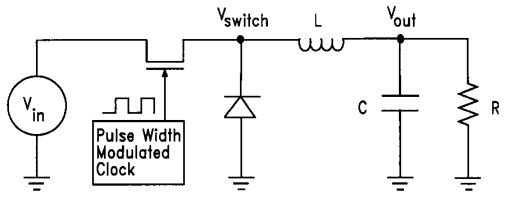

Fig. 1. Buck converter.

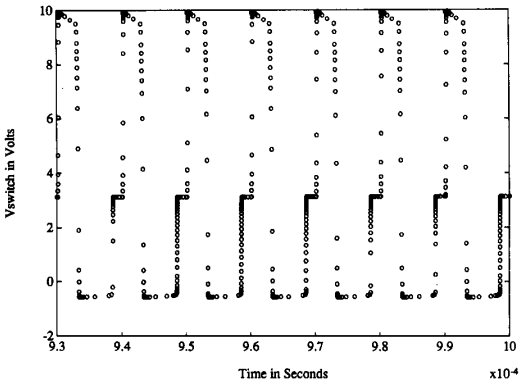

Fig. 2. Buck converter switch node voltage in steady-state.

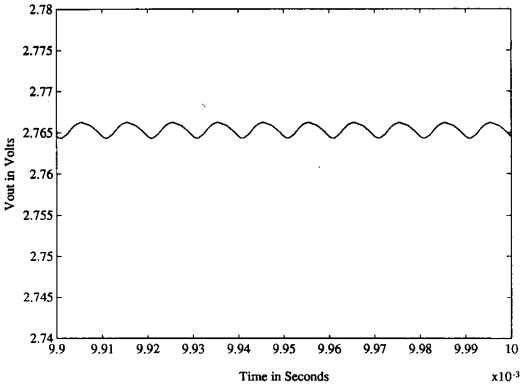

Fig. 3. Buck converter out node voltage in steady-state.

this paper is that the backward-Euler integration scheme required more than twenty timepoints for each simulated clock cycle, because of the rapid variation of the voltage at the switch node. This implies that simulating a power converter transient, which can span hundreds of clock cycles because of the low-pass filtering, will be computationally very expensive. For example, the plot in Fig. 4 is of the output voltage waveform for the power-up transient of the buck converter in Fig. 1. In this case the power-up transient is made up of more than 1000 cycles of the input clock, and the total simulation used more than 20 000 timepoints.

The number of timepoints computed during a switching converter transient simulation can be reduced by exploiting the fact that a designer typically is not interested in the details of the node voltage behavior in every clock cycle, but rather is interested in the envelope of that behavior. Specifically, we define the envelope to be a continuous function derived by interpolating the sequence formed by sampling the state every clock period T (see Fig. 5). Note that our definition of envelope is not standard. Here, the envelope is not unique given $x(t)$; the envelope generated by interpolating the sequence $x(0 + \tau), x(T + \tau), x(2T + \tau), \cdots$ depends on τ. The key advantage of considering just the envelope is that if the sequence formed by sampling the state of the beginning of each clock cycle, $x(0), x(T), x(2T), \cdots, x(mT) \cdots$, changes slowly as a function of m, the clock cycle number, it is possible to approximate the envelope without computing every clock cycle.

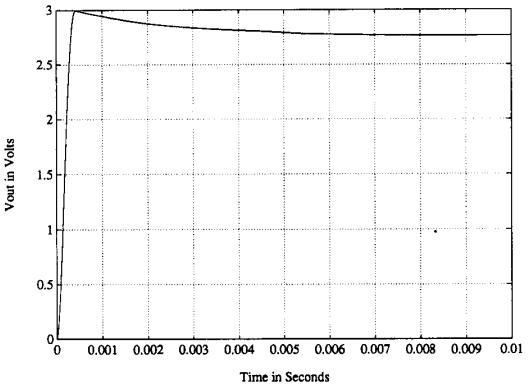

Fig. 4. Output voltage power-up transient.

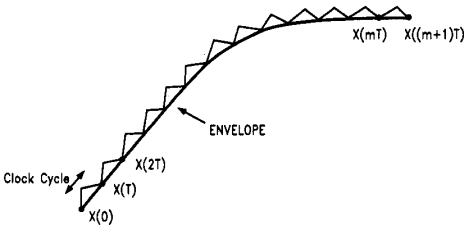

Fig. 5. Envelope definition.

III. Computing the Envelope

Most switching power converter circuits can be described by a system of differential equations of the form

$$\frac{d}{dt} p(x(t), u(t)) + f(x(t), u(t)) = 0, \tag{1}$$

where $x(t) \in \mathbb{R}^n$, the state, is the vector of capacitor voltages, and inductor currents, $u(t) \in \mathbb{R}^n$ is the vector of input sources, $p(x(t), u(t)) \in \mathbb{R}^n$ is the vector of capacitor charges and inductor fluxes, and $f(x(t), u(t)) \in \mathbb{R}^n$ is the vector of resistive currents and inductor voltages. If the state x is known at some time t_0, it is possible to solve (1) and compute the state at some later time t_1. In general, one can write

$$x(t_1) = \phi(x(t_0), t_0, t_1) \tag{2}$$

where ϕ: $\mathbb{R}^n \times \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}^n$ is a state transition function for the differential equation.

The straight-forward approach to computing the envelope of the solution to (1) is to numerically compute $x(t)$ for all t and then to sample this computed solution at $x(0), x(T), x(2T), \cdots$ to construct the envelope. If the envelope is smooth enough, then it will be possible to approximately represent an

interval of sample points, $x((m-1)T), x(mT) \cdots x((m+l)T)$ with a low order polynomial in the cycle number. For example, if over $l+1$ cycles the envelope is well approximated by a straight line, then

$$x((m+l)T) - x(mT) \approx l\big[x(mT) - x((m-1)T)\big]. \quad (3)$$

The term $[x(mT) - x((m-1)T)]$ can be thought of, imprecisely, as the derivative of the envelope at $x(mT)$ in which case (3) is loosely analogous to solving a differential equation by forward-Euler. Following that analog, l is then the cycle-step for the integration method.

To compute the envelope of a system with period T using a fixed cycle-step version of the above forward-Euler style envelope-following algorithm, a simple repetitive two-step process can be used. The first step is to compute $x(T)$, given $x(0)$, by solving (1) over the interval $[0, T]$. Then the second step is to set $x(1+l)T) = x(T) + l[x(T) - x(0)]$. This process can be repeated to compute $x((2+2l)T), x(3+3l)T)$, etc. Note that calculating the envelope over a long interval then requires solving (1) for one out of every l cycles.

Although simple to describe, a forward-Euler based approach to computing envelopes is inefficient for simulating switching converter circuits. Maintaining stability severely limits the size of the cycle-step l, just as in the standard forward-Euler algorithm. A more stable algorithm is to approximate the value of $x((m+l)T)$ by

$$x((m+l)T) - x(mT) \approx l\big[x((m+l)T) - x((m+l-1)T)\big] \quad (4)$$

which is analogous to the backward-Euler algorithm. This approach allows for larger cycle-steps than the forward-Euler based approach, but leads to a more difficult to solve equation for each cycle-step. To see this, consider computing $x(lT)$ given $x(0)$ based on (4). An $x((l-1)T)$ must be determined such that when used as an initial condition for (1), the $x(lT)$ computed with standard discretization techniques satisfies $x(lT) - x(0) = l[x(lT) - x((l-1)T)]$. This is a boundary value problem, and is in general difficult to solve. For the case of switching power or filter circuits, the above boundary value problem can be solved efficiently using a Newton or shooting method [5], and this is presented in the next section.

IV. Solution by Newton

As mentioned in the previous section, each cycle-step of a backward-Euler envelope-following algorithm applied to (1) involves finding an $x((m+l-1)T)$ which satisfies

$$x((m+l)T) - x(mT) = l\big[x((m+l)T) - x((m+l-1)T)\big] \quad (5)$$

where $x(mT)$ is known from the previous cycle-step and $x((m+l)T)$ is determined from $x((m+l-1)T)$ by solving (1) over one cycle. Using the state transition function defined in (2), the relation between $x((m+l)T)$ and $x((m+l-1)T)$ can be written as

$$x((m+l)T) = \phi\big(x((m+l-1)T), (m+l-1)T, (m+l)T\big). \quad (6)$$

Using this relation in (5) yields a nonlinear algebraic equation

$$\begin{aligned}\phi\big(x((m+l-1)T), (m+l-1)T, (m+l)T\big) - x(mT) \\ = l\big[\phi\big(x((m+l-1)T), (m+l-1)T, (m+l)T\big) - x((m+l-1)T)\big]\end{aligned} \quad (7)$$

from which $x((m+l-1)T)$ can be determined given $x(mT)$.

An iterative Newton's method can be applied to solving the above system. In general, the Newton method applied to the problem of finding an $x \in \mathfrak{R}^n$ such that $F(x) = 0$, $F: \mathfrak{R}^n \to \mathfrak{R}^n$, yields the iteration equation

$$J_F(x^k)\,[x^{k+1} - x^k] = -F(x^k), \quad (8)$$

where k is the Newton iteration count and $J_F \in \mathfrak{R}^{n\times n}$ is the Jacobian of F. Reorganizing (7) into a form to apply Newton's method leads to

$$\begin{aligned}0 &= F\big(x((m+l-1)T)\big) \\ &= \phi\big(x((m+l+1)T), (m+l-1)T, (m+l)T\big) \\ &\quad - \frac{l}{l-1}x((m+l-1)T) + \frac{1}{l-1}x(mT)\end{aligned} \quad (9)$$

In this case, J_F is given by

$$J_F(x) = \frac{\partial}{\partial x}\phi\big(x, (m+l-1)T, (m+l)T\big) - \frac{l}{l-1}I_n \quad (10)$$

where I_n is the identity matrix of size n.

The most time-consuming computation in this Newton iteration is evaluating J_F and F, which involves computing the state transition function and its derivative. The state transition function can be evaluated by numerically integrating (1) from $(m+l-1)T$ to $(m+l)T$ given $x((m+l-1)T)$. The derivative of the state transition function, referred to as the sensitivity matrix, represents the sensitivity of $x((m+l)T)$ to perturbations in $x((m+l-1)T)$ and can be computed with a small amount of additional work during the numerical integration, as is described in the following section.

V. Sensitivity Computation

To see how the computation of the state transition function and its derivative fit together, consider numerically integrating (1) with backward-Euler, which we chose for its simplicity and because it is effective for problems with rapidly varying inputs, like clocks. Given some initial time t_0 and some initial condition $x(t_0)$, applying backward-Euler to (1) results in the algebraic equation

$$g(x(t_0+h), x(t_0)) = \frac{1}{h}\big(p(x(t_0+h) - p(x(t_0))\big) + f(x(t_0+h)) = 0, \quad (11)$$

where $h \in \mathfrak{R}$ is the timestep. Note we have dropped explicitly denoting the dependence of p and f on the input u for simplicity.

Equation (11) is usually solved with Newton's method, for which the iteration equation is

$$J_g(x^{(k)}(t_0+h))\,(x^{(k+1)}(t_0+h) - x^{(k)}(t_0+h)) = -g(x^{(k)}(t_0+h), x(t_0)) \quad (12)$$

where k is the Newton iteration index, and $J_g(x(t)) \in \Re^{n \times n}$ is the Frechet derivative of the nonlinear equation in (11) and is given by

$$J_g(x(t)) = \frac{\partial g(x(t), \cdot)}{\partial x(t)} = \frac{1}{h}\frac{\partial p(x(t))}{\partial x(t)} + \frac{\partial f(x(t))}{\partial x(t)}. \quad (13)$$

Solving (11) yields an approximation $x(t_0 + h) = \phi(x(t_0), t_0, t_0 + h)$. Implicitly differentiating (11) for $x(t_0 + h)$ with respect to $x(t_0)$ yields

$$J_g(x(t_0 + h))\frac{\partial x(t_0 + h)}{\partial x(t)} = \frac{1}{h}\frac{\partial p(x(t_0))}{\partial x(t_0)}. \quad (14)$$

Given an $x(t_0)$, (11) can be repeatedly applied to approximately compute $x(t_0 + T) = \phi(x(t_0), t_0, t_0 + T)$, and (14) can be repeatedly applied to approximately compute the sensitivity matrix $\partial x(t_0 + T)/\partial x(t_0) = \partial\phi(x(t_0), t_0 t_0 + T)/\partial x(t_0)$ [1]. Note that J_g is required in both (12) and (14), and thus the sensitivity matrix update can be made very efficient by factoring J_g once and using it for both computations. For large problems, though, computing the dense $n \times n$ sensitivity matrix can become expensive.

VI. Implementation and Test Results

An envelope-following method has been implemented in the Nitswit [6] simulation program. The program is written in "C," and runs under the UNIX operating system. The program uses a trapezoidal-rule based envelope-following algorithm for which the cycle-step update equation is

$$\frac{x((m + l - 1)T) - (mT)}{l - 1} = \frac{1}{2}\left[x((m + l)T) - x((m + l - 1)T)\right] + \frac{1}{2}\left[x((m)T) - x((m - 1)T)\right] \quad (15)$$

The terms $[x((m + l)T) - x((m + l - 1)T)]$ and $[x((m)T) - x((m - 1)T)]$ in (15) can be thought of as envelope derivatives at $x((m + l)T)$ and $x(mT)$ respectively. Just as in the classical trapezoidal-rule, the average of these two derivatives is used in the cycle-step update equation. The Newton method described above is used to solve for the cycle-step update and as in standard integrators, the cycle-steps for the follower are selected automatically, based on examining both the envelope truncation error and the iteration count for the Newton method.

For the envelope-following approach to be more efficient than classical methods for a given problem, it must necessarily be possible to accurately represent the envelope of interest with a small fraction of the clock cycles. In addition, the Newton method used to solve the envelope update equations must converge rapidly, as each Newton iteration involves numerically simulating an entire converter clock cycle. If the problem is simulating an open-loop converter, by which we mean a converter where the frequency and duty cycle of the input clock are not functions of the converter state, the Newton method does converge very rapidly.

That the shooting Newton method should converge rapidly is clear for the case where an open-loop converter is constructed from clock-controlled ideal switches and other linear circuit elements. For such a converter the state transition function is affine (linear plus a constant [10] and J_F in (10) is a constant. This implies that the Newton method will always converge in one iteration. For realistic circuits in which switches are implemented by transitions and diodes, the state transition function over one cycle will still be nearly affine, and in our experience, the Newton method typically converges in three or fewer iterations at each cycle-step.

It is possible to further exploit the nearly affine property of the open-loop converter state transition function by only computing J_F for the first Newton iteration in each cycle-step. This is a significant savings, as it avoids recomputing the sensitivity matrix and usually doesn't slow the Newton method's convergence.

Algorithm: Nitswit Envelope-Follower

```
m = 0
While mT < STOPTIME
{Select the cycle-step l.
   Predict a first guess, x^0((m + l - 1)T).
   Numerically integrate (1) from (m + l - 1)T to (m +
      l)T to compute x^0((m + l)T) and ∂x^0((m +
      l)T)/∂x((m + l - 1)T).
   Compute J_F(x^0((m + l - 1)T)) as in (10)
   Set k = 0
   Until Newton Converges {
      Solve the Newton update equation for x^{k+1}((m + l
         - 1)T).
      Numerically integrate (1) from (m + l - 1)T to (m
         + l)T to compute x^k((m + l)T)
   }
   m = m + l.
}
```

Exactly how the envelope-following method behaves can be seen by examining Fig. 6, which compares two techniques for computing the power-up transient of the buck converter in Fig. 1. The continuous line is a portion of the power-up transient computed with the classical SPICE-like method, and dark segments are the results produced by the envelope-following method. As can be seen in the figure, the envelope-following method computes only occasional cycles, but the output voltage for the computed cycles are within a few percent of those computed with the classical method.

In the table below we present a comparison between the cpu time used by classical and envelope-following methods in simulating the start-up transient from three types of open-loop switching power supplies, a resonant converter res [3], and two buck converter circuits with the topology of Fig. 1: one whose steady-state is discontinuous conduction, dbuck, and one whose steady-state is continuous conduction cbuck (the converter closed is discussed below). In each case, the clocking is provided by a user-defined source. As can be seen from the table, the envelope-following method can be very efficient, particularly when the simulation interval is long compared to the clock period.

A. The Difficulty Simulating Closed-Loop Controllers

In a closed-loop converter the input clock duty cycle is a function of the converter's state, and such converters can in principle, be simulated with the envelope-following method as implemented in Nitswit. A comparison of envelope-following to classical methods for a closed-loop buck converter, closed, is given in Table I, but the results are not that encouraging.

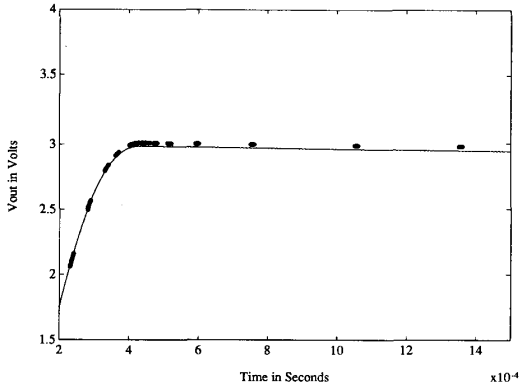

Fig. 6. Comparison of classical and envelope-following solutions.

TABLE I
CPU TIME (IN SECONDS ON A SUN4 260) COMPARISONS FOR CLASSICAL VERSUS ENVELOPE-FOLLOWING SIMULATION

Circuit	Nodes	Interval/Clock	Classical	Envelope Following
quasi	7	200	144	38.4
cbuck	4	300	131	24.6
dbuck	4	1000	551	65.9
closed	5	600	52.4	17.4

Experiments with Nitswit indicate that the obvious explanation for the poorer efficiency, that closed-loop converters have more nonlinear state transition functions, is not the dominant problem.

The difficulty simulating closed-loop loop converters is that they typically include control circuitry which produce large, very rapid responses to small changes in the converter output. That these controller variables are nearly algebraic functions of other system states implies that they are independent of their own past, and need not be envelope-followed. Eliminating these variables from the envelope computation will allow larger cycle-steps, and we are investigating automatic ways of determining nearly algebraic variables based on examining the sensitivity matrix.

VII. CONCLUSION

In this paper it is shown that an envelope-following approach to the simulation of switching power and filter circuits can provide substantial speed improvements over classical simulation methods. Several aspects of the method are still under investigation; of particular importance is finding techniques that are efficient even for closed-loop converters. Also, it has been observed that most of the entries in the sensitivity matrix remain close to zero, and ways to exploit this are being considered. In addition, the effectiveness of the envelope following is somewhat dependent on where the cycle boundaries are placed, and an automatic selection method is desirable.

ACKNOWLEDGMENT

The authors would like to thank Professor John Kassakian, Professor James Kirtley, Professor Martin Schlecht, Professor George Verghese at M.I.T.'s Laboratory for Electromagnetic and Electronic Systems for their assistance in understanding the switching power simulation problem, and Prof. A. Sangiovanni-Vincentelli and Prof. A. R. Newton at U. C. Berkeley, for their suggestions. In addition, the authors are indebted to Ken Kundert from Cadence Design Systems for his suggestions and for his contributions to the Nitswit program.

REFERENCES

[1] T. Aprille, T. Trick, "Steady-state analysis of nonlinear circuits with periodic inputs," *Proc. IEEE*, Jan. 1972.
[2] C. J. Hsaio, R. B. Ridley, H. Naitoh, and F. C. Lee, "Circuit-oriented discrete-time modeling and simulation for switching converters," *IEEE Power Electronics Specialists' Conf. Rec.*, 1987.
[3] L. F. Casey and M. F. Schlecht, "A high frequency, low volume, point-of-load power supply for distributed power systems," *IEEE Power Electronics Specialists' Conf. Rec.*, 1987.
[4] J. Kassakian, "Simulating power electronic systems—A new approach," *Proc. IEEE*, vol. 67, Oct. 1979.
[5] H. B. Keller, *Numerical Solution of Two-Point Boundary Value Problems*, Society of Industrial and Applied Mathematics, 1976.
[6] K. Kundert, J. White, and A. Sangiovanni-Vincentelli, "An envelope-following method for the efficient transient simulation of switching power and filter circuits," in *Proc. Int. Conf. Computer-Aided Design*, Santa Clara, CA, Oct. 1988, pp. 446-449.
[7] S. Leeb, "Recognition of dynamic patterns in high frequency dc-dc switching converters," MIT Department of Electrical Engineering and Computer Sciences S.M.E.E. thesis, Feb. 1989.
[8] L. W. Nagel, "SPICE2: A computer program to simulate semiconductor circuits," Electronics Research Lab Report, ERL M520, Univ. of Calif., Berkeley, May 1975.
[9] L. Petzold, "An efficient numerical method for highly oscillatory ordinary differential equations," *SIAM J. Numer. Anal.*, vol. 18, no. 3, June 1981.
[10] G. C. Verghese, M. E. Elbuluk, and J. G. Kassakian, "A general approach to sampled-data modeling for power electronic circuits," *IEEE Trans. Power Electron.*, vol. PE-1, no. 2, Apr. 1986.

Jacob White (S'80-M'83) received the B.S. degree in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, and the S.M. and Ph.D. degrees in electrical engineering and Computer Science from the University of California, Berkeley.

He worked at the IBM T. J. Watson research center from 1985 to 1987, was the Analog Devices Career Development Assistant Professor at the Massachusetts Institute of Technology from 1987 to 1989, and is a 1988 Presidential Young Investigator. His current research interests are in theoretical and practical aspects of serial and parallel numerical methods for problems in circuit design and fabrication.

Steven B. Leeb, (S'89) for a photograph and biography, see this issue, p. 302.